Clicker Questions for February 18
What is \( \text{summer '((1 2 3) (4 5 6))} \)?

A. 21 (=1+2+3+4+5+6))
B. (6 15)
C. 0
D. It causes an error
Answer D: (summer '( (1 2 3) (4 5 6))) generates an error

(define summer (lambda (L)
    (cond
        [(null? L) 0]
        [(else apply + (map summer L))])))

If you give summer a flat list such as (1 2 3) it tries to map summer onto the list, computing ( (summer 1) (summer 2) (summer 3) ) and those calls to summer crash.
I am trying to write \( (\text{sumsq } x \ y \ z \ w \ ...) \) that adds the squares of its arguments.

\[
\text{(define sumsq (lambda args}
 \text{    (cond}
 \text{      [(null? args) 0]}
 \text{      [else (+ (* (car args) (car args)) (sumsq ???))]))})
\]

What goes in place of \((\text{sumsq ???})\)? How does \text{sumsq} recurse on \((\text{cdr args})\)?

A. \((\text{sumsq (cdr args)})\)
B. \((\text{apply sumsq (cdr args)})\)
C. \((\text{map sumsq (cdr args)})\)
D. \((\text{cadr args})\)
Answer B: (apply sumsq (cdr args))